APPARATUS AND METHOD FOR DISTRIBUTED MEMORY CONTROL IN A GRAPHICS PROCESSING SYSTEM

TECHNICAL FIELD

The present invention is related generally to the field of computer graphics, and more particularly, to a memory system for use in a computer graphics processing system.

BACKGROUND OF THE INVENTION

A heterogeneous memory system is a memory system where several different memories, or levels of memory, are used to satisfy memory demands of an computer application. An example of an application for a heterogeneous memory system is in graphics processing systems. Different levels of memory are used by a graphics processing system to facilitate graphics processing and rendering of graphics images on a display. A first level of memory is typically embedded memory that is fabricated directly on the same semiconductor substrate as a graphics processor. Embedded memory can provide data to the graphics processor at very low access times, and consequently, increase the speed at which graphics data may be processed. A second level of memory is typically memory that is external to the device, but located on the same graphics card as the graphics processor. Memory such as this is commonly referred to as external, or local memory. A third level of memory is AGP memory, or host memory that the graphics processor can access through a system bus. Host memory generally has the greatest access time of the three levels of memories because the graphics processor can only access the AGP memory via a system bus and several different memory and bus controllers. Although local memory can provide data more quickly than the host memory, it still is considerably slower than the embedded memory of the first level of memory.

For a conventional heterogeneous memory system, there are two typical arrangements. A first example of a heterogeneous memory system is arranged with a single memory controller to handle all memory accesses. Such an arrangement is

illustrated in Figure 1. The memory system 10 includes a central memory controller 12 coupled to both memory 20 through memory bus 16, and memory 22 through memory bus 18. The memory 20 may be representative of embedded memory, and the memory 22 may be representative of external memory. In operation, the central memory controller 12 receives memory access requests from various requesting entities, such as a graphics processor, over buses 14a-n. The central memory controller 12 services the various memory access requests by determining whether the requested memory address is located in the memory 20 or the memory 22. The appropriate memory device is accessed and data is written to or read therefrom. An arrangement such as memory system 10 has the advantage that additional memory may be easily added because all memory access requests are serviced by the central memory controller 12. For the same reason, the various memory access requests can all be handled seamlessly by the central memory controller 12. That is, when a memory access request is made, only the central memory controller 12 must determine which memory, either memory 20 or memory 22, to access. However, a problem with the arrangement of memory system 10 is that there are physical limitations as to the number of buses 14a-n that may be routed to the memory controller 12. Additionally, as the complexity of the central memory controller 12 increases to accommodate a greater number of memory access requests, the amount of space the central memory controller occupies also increases. Thus, space overhead issues become a concern in applications where small graphics processing systems are desired.

A second example of a heterogeneous memory system is shown in Figure 2. Memory system 30 addresses some of the concerns raised by the memory system 10 of Figure 1. The memory system 30 includes a central memory controller 12 coupled to a memory 20 through a memory bus 16. The central memory controller 12 services only the memory access requests made to memory 20. The memory system 30 also includes memory 22 directly coupled to a requesting entity through memory bus 32. Thus, memory access requests to memory 22 may be only made over the memory bus 32. The memory 20 may represent embedded memory, while the memory 22 may represent local memory. As illustrated in Figure 2, all memory access requests to

memory 20 are controlled by the central memory controller 12. However, access to the memory 22, is controlled directly by the requesting entity coupled to the bus 32. That is, access to memory 22 can be made only by the requesting entity hardwired to the bus 32.

The memory system 30 does, to some degree, resolve the issues with regards to the physical limitations of routing a plurality of request lines to a single central memory controller, as well as space overhead issues resulting from the complexity of using a central memory controller. However, a problem with the memory system 30 is that the allocation of available memory is fixed according to the design of the circuitry. That is, the memory 22 may be accessed only by the requesting entity to which it is coupled through bus 32. Any available memory in the memory 22 cannot be reallocated for another purpose, such as storing overflow data from the memory 20. Furthermore, memory access requests must be delegated prior to being made either to the central memory controller 12 or the memory 22, rather than having all memory access requests simply handled by a single central memory controller. Moreover, adding additional memory to the memory system 30 is made more difficult by the fixed arrangement. Additional memory cannot simply be reallocated, but must be added to supplement either memory 20 or memory 22, but not both.

Therefore, there is a need for a memory system where the number of memory access request lines to a memory controller is reduced and where the available memory may be allocated efficiently.

SUMMARY OF THE INVENTION

The present invention relates to a distributed memory controller memory system for a graphics processing system having addressable memory areas, each of which is coupled to a respective memory controller. Each memory controller accesses the addressable memory area to which it is coupled. The memory controllers are further coupled to each other through a memory controller bus upon which a memory access request and data may be passed from one memory controller to other memory controller. A memory access request to a memory location in one addressable memory area, but

received by a memory controller coupled to another addressable memory area, is passed through the memory controller bus from the receiving memory controller to the memory controller coupled to the addressable memory area in which the requested location is located in order to service the memory access request. Additional addressable memory areas coupled to a respective memory controller may also be included in the memory system. The additional memory controllers are also coupled to the memory controller bus in order to receive and pass memory access requests from the other memory controllers. The addressable memory locations may be defined by values stored in registers in the respective memory controller in order for the memory controller to determine whether the requested location is within the memory area to which it is coupled.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram of a conventional heterogeneous memory system.

Figure 2 is a block diagram of an alternative conventional heterogeneous memory system.

Figure 3 is a block diagram of a computer system in which embodiments of the present invention are implemented.

Figure 4a is a block diagram of a memory system having a distributed memory controller arrangement according to an embodiment of the present invention.

Figure 4b is a block diagram of a memory system having a distributed memory controller arrangement according to another embodiment of the present invention.

Figure 5 is a block diagram of a graphics processing system including a distributed memory controller arrangement according to another embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

Embodiments of the present invention provide for a distributed memory controller arrangement that may be substituted for a memory system having a conventional central memory controller arrangement. Multiple memory controllers are arranged such that each memory controller is coupled to at least one addressable memory area which is accessible by the memory controller to which the addressable memory area is coupled. Each memory controller receives direct memory access requests from distinct requesting entities. The multiple memory controllers are coupled together by a memory controller bus, upon which data and indirect memory access requests may be passed from one memory controller to another if the requested address is outside of the addressable memory area to which the memory controller receiving the direct request is coupled.

Certain details are set forth to provide a sufficient understanding of the invention. However, it will be clear to one skilled in the art that the invention may be practiced without these particular details. In other instances, well-known circuits, control signals, timing protocols, and software operations have not been shown in detail in order to avoid unnecessarily obscuring the invention.

Figure 3 illustrates a computer system 40 in which embodiments of the present invention are implemented. The computer system 40 includes a processor 42 coupled to a host memory 44 through a memory/bus interface 46. The memory/bus interface 46 is coupled to an expansion bus 48, such as an industry standard architecture (ISA) bus or a peripheral component interconnect (PCI) bus. The computer system 40 also includes one or more input devices 50, such as a keypad or a mouse, coupled to the processor 42 through the expansion bus 48 and the memory/bus interface 46. The input devices 50 allow an operator or an electronic device to input data to the computer system 40. One or more output devices 52 are coupled to the processor 42 to provide output data generated by the processor 42. The output devices 52 are coupled to the processor 42 through the expansion bus 48 and memory/bus interface 46. Examples of output devices 52 include printers and a sound card driving audio speakers. One or more data storage devices 54 are coupled to the processor 42 through the memory/bus

interface 46 and the expansion bus 48 to store data in, or retrieve data from, storage media (not shown). Examples of storage devices 54 and storage media include fixed disk drives, floppy disk drives, tape cassettes and compact-disc read-only memory drives.

The computer system 40 further includes a graphics processing system 100 coupled to the processor 42 through the expansion bus 48 and memory/bus interface 46. Optionally, the graphics processing system 100 may be coupled to the processor 42 and the host memory 44 through other types of architectures. For example, the graphics processing system 100 may be coupled through the memory/bus interface 46 and a high speed bus 56, such as an accelerated graphics port (AGP), to provide the graphics processing system 100 with direct memory access (DMA) to the host memory 44. That is, the high speed bus 56 and memory bus interface 46 allow the graphics processing system 100 to read from and write to the host memory 44 without the intervention of the processor 42. Thus, data may be transferred to, and from, the host memory 44 at transfer rates much greater than over the expansion bus 48. A display 58 is coupled to the graphics processing system 100 to display graphics images. The display 58 may be any type of display, such as a cathode ray tube (CRT), a field emission display (FED), a liquid crystal display (LCD), or the like, which are commonly used for desktop computers, portable computers, and workstation or server applications.

Figure 4 illustrates a memory system 200 according to an embodiment of the present invention. The memory system 200 includes separate memory controllers 202, 222, 242, and 262. Each of the memory controllers 202, 222, 242, and 262 controls and accesses a respective memory 212, 232, 252, and 272 through a memory bus that couples the memory controller to a memory. The memory controllers 202, 222, 242, and 262 are also coupled to each other through a memory controller bus 216. Memory access requests, as well as data, may be transferred through the memory controller bus 216 from one memory controller to another.

Each of the memory controllers 202, 222, 242, and 262 is also coupled to a set of memory access request lines 208a-d on which the respective memory controller directly receives memory access requests. A memory controller receives direct memory

access requests from those requesting entities coupled to its particular request lines. For example, the memory controller 202 will receive direct memory access requests over the memory access request lines 208a. In determining which requesting entities a particular memory controller should receive memory access requests, factors such as physical proximity of the requesting entity to a memory controller, the memory device which a requesting entity is most likely to access, and desired access speed may be considered. As will be discussed in greater detail below, indirect memory access requests can be made by one memory controller to another through the memory controller bus 216 if the requested address is not in the addressable memory area of the memory to which the memory controller receiving the direct memory access request is coupled.

Included in each memory controller 202, 222, 242, and 262 are a respective start address register (SAR) 204a-d and a respective memory size register 206a-d (MSR). With respect to the memory controller 202, the SAR 204a stores the start address of the addressable memory area of the memory 212, and the MSR 206a stores the size or the amount of available addressable memory area of the memory 212. Similarly, the remaining SARs 204b-d and MSRs 206b-d store the respective start addresses and sizes of the addressable memory area for the memory to which the memory controller is coupled. The values stored in the SARs and MSRs of the memory controllers may be programmed by an graphics application executing on the host processor 42 (Figure 3) or, as will be described later, a graphics processor that is designed to perform graphics functions. The graphics application may update the values stored in the SARs and MSRs during execution in order to reallocate the addressable memory area. By storing the start address and size for the addressable area which each memory controller 202, 222, 242, and 262 controls, a memory controller can determine whether a direct memory access request it receives should be passed to another memory controller if the requested address is not within the range of the memory to which the memory controller receiving the direct memory access request is coupled.

Although the memory system 200 has been described as storing the start address and the amount of available addressable memory area for a memory, it will be

appreciated that other values can be used to define the memory as well, such as, the start address and the end address of an addressable memory area. Thus, the particular type of values that are stored by the memory controllers to define the addressable memory area to which it is coupled are details that may be changed, but the resulting memory system will still remain within the scope of the present invention.

The following description of the operation of the memory system 200 is provided merely by way of an example, and should not be interpreted as limiting the scope of the invention. A person of ordinary skill in the art will appreciate that some of the details of the following example, such as the start addresses and size of the addressable memory area, have been selected merely for the purposes of the following example.

In the present example, the values programmed and stored in the SARs and MSRs for the memory controller 202 are 0000 and 1000, for the memory controller 222 are 1000 and 1000, for the memory controller 242 are 2000 and 2000, and for the memory controller 262 are 4000 and 3000. A direct memory access request is received by the memory controller 222 to access memory address 1A00. Based on the values stored in the SAR 204b and MSR 206b, that is, 1000 and 1000, respectively, the memory controller 222 determines that the requested address 1A00 is within the addressable memory area of the memory 232, and services the direct memory access request.

Another direct memory access request is received by the memory controller 222, but this time it is to access memory address 0C00. The memory controller 222 determines that the requested address is not within the addressable memory area of the memory 232 and must make an indirect memory access request to another memory controller in order to service the memory access request. The requested address is less than the 1000 value stored in the SAR 204b, and consequently, the memory controller 222 passes an indirect memory access request to a memory controller having a lower starting memory address through the memory controller bus 216, namely, to the memory controller 202. The memory controller 202 receives the indirect memory access request and determines whether the memory address of the

indirect memory access request, namely 0C00, is within the addressable memory area of the memory 212. Based on the values stored in the SAR 204a and the MSR 206a, that is 0000 and 1000, respectively, the memory controller 202 determines that the address 0C00 is within the memory 212, and consequently services the memory access request. If the memory access request is a read command, then the memory controller 202 accesses the requested address, retrieves the data, and passes the data back to the memory controller 222. The memory controller 222 then completes the direct memory access request by providing the data read from the memory 212 by the memory controller 202 to the requesting entity. If the memory access request is a write command, the data is provided to the memory controller 202 along with the requested address through the memory controller bus 216 and is written into the memory 212 by the memory controller 202.

The distributed memory controller arrangement of the memory system 200 addresses the potential problem with physical limitations of the number of memory access request lines that may be routed to a central memory controller by dividing the total number of memory access request lines among different controllers. Thus, the number of memory access request lines to any one memory controller is reduced. Furthermore, the available memory of memories 212, 232, 252, and 272 may be reallocated if desired, and any memory added to the memory system 200 may be utilized in an efficient manner by changing the values stored in the SARs and MSRs of the memory controllers.

A memory system 201 according to another embodiment of the present invention is shown in Figure 4b. The memory system 201 is similar to the memory system 200 of Figure 4a, except that memory controller 243 is coupled to both memories 253 and 259. Although the memory system 200 (Figure 4a) is arranged such that there is a one-to-one correspondence between memory controllers 202, 222, 242, and 262, and a respective memory 212, 232, 252, and 272, the memory system 201 is arranged such that more than one memory device coupled to a single memory controller. The operation of the memory system 201 is generally the same as for the memory system 200, except that the value stored in the MSR 207c should span the

combined size of memories 253 and 259. In this way, the memory controller 243 is able to recognize memory access requests for both the memories 253 and 259.

Illustrated in Figure 5 is another embodiment of the present invention. A memory system similar to the memory system 200 (Figure 4a) is used in the context of the graphics processing system 100 (Figure 3). The graphics processing system 100 includes circuitry for performing various three-dimensional (3D) graphics function. As shown in Figure 5, a bus interface 302 couples the graphics processing system 100 to the expansion bus 48. In the case where the graphics processing system 100 is coupled to the processor 42 and the host memory 44 through the high speed data bus 56 and the memory/bus interface 46, the bus interface 302 will include a DMA controller (not shown) to coordinate transfer of data to and from the host memory 44 and the processor 42. A graphics processor 308 is coupled to the bus interface 302 and is designed to perform various graphics and video processing functions, such as, but not limited to, generating vertex data and performing vertex transformations for polygon graphics primitives that are used to model 3D objects. In a preferred embodiment, the graphics processor 308 is a reduced instruction set computing (RISC) microprocessor. The graphics processor 308 is coupled to a triangle engine 312 that includes circuitry for performing various graphics functions, such as clipping, attribute transformations, rendering of graphics primitives, and generating texture coordinates from a texture map.

A pixel engine 318 is coupled to receive the graphics data generated by the triangle engine 312. The pixel engine 318 contains circuitry for performing various graphics functions, such as, but not limited to, texture application or mapping, bilinear filtering, fog, blending, and color space conversion. Texture mapping refers to techniques for adding surface detail, or a texture map, to areas or surfaces of polygons used to model the 3D objects. After the texture mapping process, a version of the texture image is visible on surfaces of the polygon with the proper perspective. A typical texture map includes point elements ("texels") which reside in a texture coordinate space is stored in the host memory 44 of the computer system 40. A portion of the texture map that is currently being applied by the pixel engine 318 is stored in a texture cache 324 for quick access during texture processing. A display controller 332

coupled to pixel engine 318 controls the transfer of destination color values from the pixel engine 318 to a FIFO 336. Destination color values stored in the FIFO 336 are provided to a display driver 340 that includes circuitry to provide digital color signals, or convert digital color signals to red, green, and blue analog color signals, to drive the display 58 (Figure 3).

Also included in the graphics processing system 100 is a distributed memory controller arrangement similar to the memory system 200 of Figure 4a. That is, instead of a conventional memory system having a central memory controller to service all memory access requests, the graphics processing system 100 uses a memory system where the responsibility of servicing the memory access requests is distributed among multiple memory controllers 202, 222, 242, and 262, coupled to a respective memory 212, 232, 252, and 44 (Figure 3), and linked together through a memory controller bus 216. The operation of a distributed memory controller arrangement has been previously described with respect to Figure 4.

Each of the memory controllers receives direct memory access requests from a respective circuit block to access the memory to which the memory controller is coupled. The arrangement of the memory controllers is based in part, as mentioned previously, the proximity of the memory and memory controller to a requesting entity, the desired access time, as well as the type of memory which the requesting entity is likely to access frequently.

In the graphics processing system 100, the memories 212 and 232 may be embedded memory fabricated on the same semiconductor substrate as the graphics processor 308, triangle engine 312, and pixel engine 318. Memory access times for memory access requests made by the triangle engine 312 and the pixel engine 318 will be relatively short because of the proximity of the embedded memories 212 and 232, which will facilitate fast graphics processing. The memory 252 may be implemented by external or local memory, which is, as mentioned previously, memory which is located with the graphics processing system 100, but is not fabricated on the same substrate as the graphics processing circuit blocks. Typically, local memory is implemented using random access memory (RAM), such as dynamic access memory (DRAM), or static

random access memory (SRAM), located on the same graphics card as the graphics processor. Although the access time of the memory 252 is greater than for the embedded memories 212 and 232, it is still shorter than for the host memory 44 (Figure 3). The rate at which texture data is provided to the pixel engine 318 is improved by the presence of the texture cache 324. That is, as mentioned previously, a subset of the texture data presently used for texture application is stored in the texture cache for fast access. The memory controller 262 is coupled to the host memory 44 (Figure 3). Although the host memory 44 has the longest access time, it does have the benefit of having the greatest available addressable memory area. Graphics data that is not immediately needed by one of the processing blocks, or data that may be needed at a later time, may be stored in the host memory 44.

From the foregoing it will be appreciated that, although specific embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without deviating from the spirit and scope of the invention. Accordingly, the invention is not limited except as by the appended claims.